CLAIM AMENDMENTS

1. (Original) A method for aiding debugging of program code in a computer system, the computer system comprising an input system and an output system, the program code comprising a plurality of program code statements, the method comprising:

utilizing the input system to indicate an error variable in the program code, the error variable containing an error value that differs from a first desired value;

obtaining an error set of the error variable, the error set comprising a subset of the program code statements, each program code statement in the error set being relationally connected to the error variable;

assigning a priority value to each program code statement in the error set, each priority value indicating a computed probability that the associated program code statement is an error source of the error variable; and

utilizing the output system to present each program code statement in the error set in a manner that indicates the relative ordering of the priority value of the program code statement to the other priority values of the other program code statements in the error set.

2. (Original) The method of claim 1 wherein the program code statements in the error set are presented in order from a most-likely error source to a least-likely error source.

3. (Original) The method of claim 1 wherein each program code statement in the error set is presented with the associated priority value.

4. (Currently Amended) The method of claim 1 wherein the program code is repetitively executed by way of a plurality of execution cycles, ~~and obtaining the~~ <u>and obtaining the</u> error set comprises:

obtaining an error cycle that is an execution cycle in which the error variable obtains the error value;

obtaining an execution set comprising all program code statements that are executed in the error cycle; and utilizing the program code statements in the execution set to generate the error set, every program code statement in the error set being in the execution set and also being relationally connected to the error variable.

5. (Previously Presented) The method of claim 4 wherein the program code comprises a correct variable, the correct variable having a value that agrees with a second desired value in the error cycle, and wherein the step of assigning the priority value to each program code statement in the error set comprises:

for a first execution cycle prior to the error cycle, obtaining a first sensitized set for the correct variable, the first sensitized set comprising at least a program code statement that is relationally connected to the correct variable, that is executed in the first execution cycle, and that is in the error set; and

for each program code statement in the first sensitized set, applying a scaling function to the priority value associated with the identical program code statement in the error set, the scaling function setting a reduced computed probability that the associated program code statement in the program code is the error source of the error variable.

6. (Original) The method of claim 5 wherein the scaling function adds a constant value to the I priority value associated with the identical program code statement in the error set.

7. (Original) The method of claim 5 wherein a second sensitized set is additionally obtained for the error variable, the second sensitized set comprising at least a program code statement that is relationally connected to the error variable, that is executed in the first execution cycle, and that is in the error set, and for each program code statement in the second sensitized set, the scaling function is applied to the priority value associated with the identical program code statement in the error set.

8. (Original) The method of claim 5 wherein a plurality of execution cycles prior to the error cycle are utilized to assign the priority values.

9. (Original) The method of claim 5 wherein a plurality of correct variables are utilized to assign the priority values.

10. (Original) The method of claim 1 wherein the program code is hardware development language (HDL) code.

11. (Original) A method for aiding debugging of program code in a computer system, the computer system comprising an input system and an output system, the program code comprising a plurality of program code statements that are repetitively executed by way of a plurality of execution cycles, the method comprising:

utilizing the input system to indicate an error variable in the program code, the error variable containing an error value that differs from a first desired value;

obtaining an error cycle that is an execution cycle in which the error variable obtains the error value;

obtaining an execution set comprising all program code statements that are executed in the error cycle;

utilizing the program code statements in the execution set to generate an error set of program code statements, every program code statement in the error set being in the execution set and also being relationally connected to the error variable; and

utilizing the output system to present each statement in the error set.

12. (Original) The method of claim 11 further comprising assigning a priority value to each statement in the error set, each priority value indicating a computed probability that the associated program code statement in the error set is an error source of the error variable, and each program code statement in the error set presented via the output system in a manner that indicates the relative ordering of the priority value of the program code statement to the other priority values of the other program code statements in the error set.

13. (Original) The method of claim 12 wherein the statements in the error set are presented in order from a most-likely error source to a least-likely error source.

14. (Original) The method of claim 12 wherein each statement in the error set is presented with the associated priority value.

15. (Previously Presented) The method of claim 12 wherein the program code comprises a correct variable, the correct variable having a value that agrees with a second desired value in the error cycle, and wherein the step of assigning the priority value to each program code statement in the error set comprises:

for a first execution cycle prior to the error cycle, obtaining a first sensitized set for the correct variable, the first sensitized set comprising at least a program code statement that is relationally connected to the correct variable, that is executed in the first execution cycle, and that is in the error set; and

for each program code statement in the first sensitized set, applying a scaling function to the priority value associated with the identical program code statement in the error set, the scaling function setting a reduced computed probability that the identical program code statement is the error source of the error variable.

16. (Original) The method of claim 15 wherein the scaling function adds a constant value to the priority value associated with the related program code statement in the error set.

17. (Original) The method of claim 15 wherein a second sensitized set is additionally obtained for the error variable, the second sensitized set comprising at least a program code statement that is relationally connected to the error variable, that is executed in the first execution cycle, and that is in the error set, and for each program code statement in the second sensitized set, the scaling function is applied to the priority value associated with the identical program code statement in the error set.

18. (Original) The method of claim 15 wherein a plurality of execution cycles prior to the error cycle are utilized to assign the priority values.

19. (Original) The method of claim 15 wherein a plurality of correct variables are utilized to assign the priority values.

20. (Original) The method of claim 11 wherein the program code is hardware development language (HDL) code.

21. (Original) A computer system comprising:

a processor;

an output system for presenting information to a user;

an input system for obtaining data from the user; and

a memory for storing code and data for the processor, the memory comprising:

program code comprising a plurality of program code statements;

debug information about the program code;

an execution system for generating the debug information; and

a prioritizing system, executed by the processor, that utilizes the debug information to perform the following:

utilizing the input system to indicate an error variable in the program code, the error variable containing an error value that differs from a first desired value;

obtaining an error set of the error variable, the error set comprising a subset of the program code statements, each program code statement in the error set being relationally connected to the error variable;

assigning a priority value to each program code statement in the error set, each priority value indicating a computed probability that the associated program code statement is an error source of the error variable; and

utilizing the output system to present each program code statement in the error set in a manner that indicates the relative ordering of the priority value of the program code statement to the other priority values of the other program code statements in the error set.

22. (Original) The computer system of claim 21 wherein the program code statements in the error set are presented in order from a most-likely error source to a least-likely error source.

23. (Original) The computer system of claim 21 wherein each program code statement in the error set is presented with the associated priority value.

24. (Original) The computer system of claim 21 wherein the program code is repetitively executed by the execution system in a plurality of execution cycles, and obtaining the error set comprises:

obtaining an error cycle that is an execution cycle in which the error variable obtains the error value;

obtaining an execution set comprising all program code statements that are executed in the error cycle; and

utilizing the program code statements in the execution set to generate the error set, every program code statement in the error set being in the execution set and also being relationally connected to the error variable.

25. (Previously Presented) The computer system of claim 24 wherein the program code comprises a correct variable, the correct variable having a value that agrees with a second desired value in the error cycle, and wherein the step of assigning the priority value to each program code statement in the error set comprises:

for a first execution cycle prior to the error cycle, obtaining a first sensitized set for the correct variable, the first sensitized set comprising at least a program code statement that is relationally connected to the correct variable, that is executed in the first execution cycle, and that is in the error set; and

for each program code statement in the first sensitized set, applying a scaling function to the priority value associated with the identical program code statement in the error set, the scaling function setting a reduced computed probability that the associated program code statement in the program code is the error source of the error variable.

26. (Original) The computer system of claim 25 wherein the scaling function adds a constant value to the priority value associated with the identical program code statement in the error set.

27. (Original) The computer system of claim 25 wherein a second sensitized set is additionally obtained for the error variable, the second sensitized set comprising at least a program code statement that is relationally connected to the error variable, that is executed in the first execution cycle, and that is in the error set, and for each program code statement in the second sensitized set, the scaling function is applied to the priority value associated with the identical program code statement in the error set.

28. (Original) The computer system of claim 25 wherein a plurality of execution cycles prior to the error cycle are utilized to assign the priority values.

29. (Original) The computer system of claim 25 wherein a plurality of correct variables are utilized to assign the priority values.

30. (Original) The computer system of claim 21 wherein the program code is hardware development language (HDL) code.